AMENDMENT TO THE CLAIMS

Please amend the claims as follows. The claims are in the format as required by 35 C.F.R. § 1.121. For the convenience of the Examiner, each independent claim is presented on a separate sheet.

1. (Currently amended) A computer-implemented method of deploying ~~a component~~ components of a site between systems, the method comprising the steps of:

~~designating a~~ storing assets of at least one component of ~~the~~ a site ~~intended~~ designated for export~~; collecting at least one object of the component in~~ as an individual export file, wherein the ~~at least one~~ site comprises a collection of software objects manipulatable by a set of users having assigned privileges defined by permissions associated with each software object in the collection of software objects ~~includes at least one non-file asset configured to operate on a system at the remote location~~;

transferring the individual export file to ~~the~~ a system at a remote location, wherein the system is a web portal capable of executing program instructions; and

extracting ~~each object~~ the assets from the individual export file to a ~~location~~ plurality of locations on the system at the remote location, wherein the assets include file assets and non-file assets configured to operate on the web portal at the remote location.

2. (Cancelled).

3. (Currently amended) The method according to claim 1, ~~wherein designating the component for export comprises~~ further comprising querying a local system to identify one or more components associated with the site.

4. (Previously Presented) The method according to claim 1, wherein each non-file asset is an extensible markup language fragment with a predetermined structure.

5. (Original) The method according to claim 4, further comprising parsing the extensible markup language fragment.

6. (Previously Presented) The method according to claim 5, further comprising instantiating each at least one non-file asset.

7. (Previously Presented) The method according to claim 1, wherein each non-file asset includes at least one of: a permissions object, a user preference object, a settings object, a menu object, an instantiated programming language object and a user group object.

8-10. (Canceled).

11. (Currently amended) A computer-implemented method of importing components of a site to a system at a remote location in a portal framework, the method comprising the steps of:

extracting at least one object of a component from ~~the~~ an individual export file of an exported site, wherein the exported site comprises a collection of software objects manipulatable by a set of users having assigned privileges defined by permissions associated with each software object in the collection of software objects;

wherein the at least one object includes at least one non-file asset configured to operate on the system at the remote location; and

storing each object of the component to a location on the system at the remote location.

12. (Previously Presented) The method according to claim 11, further comprising transferring the individual export file to the system at the remote location.

13. (Previously Presented) The method according to claim 11, further comprising parsing each non-file asset wherein each non-file asset is constructed as an extensible markup language fragment with a predetermined structure.

14. (Previously Presented) The method according to claim 11, further comprising instantiating each non-file assets.

15. (Previously Presented) The method according to claim 11, wherein each non-file asset includes at least one of: a permissions object, a user preference object, a settings object, a menu object, an instantiated programming language object and a user group object.

16. (Currently amended) A computer program product embodied in a computer readable medium for deploying a component of a site between systems in a portal framework, the computer ~~program product comprising: a computer readable medium; and~~ readable medium carrying computer program instructions, ~~recorded on the computer readable medium,~~ executable by a processor, for performing the ~~step~~ steps of:

designating a component of the site intended for export, wherein the site comprises a collection of software objects manipulatable by a set of users having assigned privileges defined by permissions associated with each software object in the collection of software objects;

collecting at least one object of the component in an individual export file;

wherein the at least one object includes at least one non-file asset configured to operate on the system at the remote location;

transferring the individual export file to a system at a remote location; and

extracting each object from the individual export file to a location on the system at the remote location.

17. (Previously Presented) The computer program product according to claim 16, further comprising computer program instructions for performing the step of collecting the assets of the component.

18. (Previously Presented) The computer program product according to claim 17, wherein designating the component for export further comprises querying a local system.

19. (Previously Presented) The computer program product according to claim 17, wherein each non-file asset is constructed as an extensible markup language fragment with a predetermined structure.

20. (Previously Presented) The computer program product according to claim 19, further comprising computer program instructions for performing the step of parsing each extensible markup language fragment.

21. (Previously Presented) The computer program product according to claim 20, further comprising computer program instructions for performing the step of instantiating each non-file assets.

22. (Previously Presented) The computer program product according to claim 16, wherein each non-file assets include at least one of: a permissions object, a user preference object, a settings object, a menu object, an instantiated programming language object and a user group object.

23-25. (Canceled).

26. (Currently amended) A computer program product embodied in a computer readable medium for importing components of a site to a system at a remote location in a portal framework, the computer ~~program product comprising the steps of: a computer readable medium; and~~ readable medium carrying computer program instructions, ~~recorded on the computer readable medium,~~ executable by a processor, for performing the ~~step~~ steps of:

extracting at least one object of a component from the individual export file of an exported site wherein the at least one object includes at least one non-file asset configured to operate on the system at the remote location, wherein the exported site comprises a collection of software objects manipulatable by a set of users having assigned privileges defined by permissions associated with each software object in the collection of software objects; and

storing each object of the component to a location on the system at the remote location.

27. (Previously Presented) The computer program product according to claim 26, further comprising transferring the individual export file to the system at the remote location.

28. (Previously Presented) The computer program product according to claim 26, further comprising instantiating each non-file asset.

29. (Previously Presented) The computer program product according to claim 26, wherein each non-file asset includes at least one of: a permissions object, a user preference object, a settings object, a menu object, an instantiated programming language object and a user group object.

30. (Currently amended) A system for deploying a component of a site between systems in a portal framework, comprising:

~~a first deployment manager on~~ a local system configured with a first deployment manager and operable to collect at least one object of a component of a site for deployment as an individual export file, wherein each object includes at least one non-file asset configured to operate on a system at a remote location;

a means for transferring the ~~group~~ individual export file from the local system to ~~a~~ the system at ~~a~~ the remote location; and

~~a second deployment manager on~~ the system at the remote location configured with a second deployment manager and operable to extract each object from the individual export file to a location on the system at the remote location.

31. (Currently amended) The system according to claim 30, wherein the object is collected by a first ~~deployment manager~~ component module.

32. (Original) The system according to claim 31, wherein the first component module is further operable to query the local system to identify the component designated for export.

33. (Canceled).

34. (Previously Presented) The system according to claim 31, wherein the first component module is further operable to construct each non-file asset as an extensible markup language fragment with a predetermined structure.

35. (Previously Presented) The system according to claim 34, wherein the second deployment manager further includes a second component module operable to parse each extensible markup language fragment.

36. (Previously Presented) The system according to claim 34, wherein the second component module is further operable to instantiate each non-file assets.

37. (Previously Presented) The system according to claim 30, wherein the second component module is further operable to extract each object of the component from the export file to a location on the system at the remote location.

38. (Previously Presented) The system according to claim 30, wherein the non-file assets include at least one of: a permissions object, a user preference object, a settings object, a menu object, an instantiated programming language object and a user group object.

39-42. (Cancelled).

43. (Currently amended) A system for importing a component of a site ~~from a system at a remote location~~ in a portal framework comprising:

a local system configured with a component module operable to:

extract at least one object of the component of the site from ~~the~~ an individual export file, wherein each object includes at least one non-file asset configured to operate on ~~a system at a~~ the local system, wherein the site comprises a collection of software objects manipulatable by a set of users having assigned privileges defined by permissions associated with each software object in the collection of software objects, and wherein at least one non-file asset is constructed as an extensible markup language fragment having a predetermined structure;

parse ~~an~~ the extensible markup language fragment ~~including at least some of the assets~~;

instantiate each non-file asset; and

store each object of the component to the local system.

44-45. (Canceled).

46. (Previously Presented) The system according to claim 43, wherein each non-file asset includes at least one of: a permissions object, a user preference object, a settings object, a menu object, an instantiated programming language object and a user group object.

47-58. (Canceled).

59. (Previously Presented) The method according to claim 1, further comprising collecting at least one file-asset of the component in the individual export file, wherein each file-asset is configured to operate on the system at the remote location.

60. (Currently amended) The method according to claim 59, ~~wherein designating the component comprises~~ further comprising querying a subsystem to identify one or more components associated with the site.

61. (Previously Presented) The method according to claim 60, wherein the subsystem is configured to manage a type of component, wherein the type includes at least one of a site, module, template and style.

62. (Previously Presented) The method according to claim 61, wherein the subsystem collects each object and each file-asset.

63. (Previously Presented) The method according to claim 11, further comprising

extracting at least one file-asset of the component from the individual export file; and

storing each file-asset to a location on the system at the remote location.

64. (Previously Presented) The method according to claim 63, wherein storing each object and file-asset is accomplished by a subsystem at the remote location, wherein the subsystem is configured to manage a type of component, wherein the type includes at least one of a site, module, template and style.

65. (Previously Presented) The computer program product according to claim 16, further comprising computer program instructions for collecting at least one file-asset of the component in the individual export file, wherein each file-asset is configured to operate on the system at the remote location.

66. (Previously Presented) The computer program product according to claim 65, wherein designating the component comprises querying a subsystem.

67. (Previously Presented) The computer program product according to claim 66, wherein the subsystem is configured to manage a type of component, wherein the type includes at least one of a site, module, template and style.

68. (Previously Presented) The computer program product according to claim 67, wherein the subsystem collects each object and each file-asset.

69. (Previously Presented) The computer program product according to claim 26, further comprising

extracting at least one file-asset of the component from the individual export file; and storing each file-asset to a location on the system at the remote location.

70. (Previously Presented) The computer program product according to claim 69, wherein storing each object and file-asset is accomplished by a subsystem at the remote location, wherein the subsystem is configured to manage a type of component, wherein the type includes at least one of a site, module, template and style.

71. (Previously Presented) The system according to claim 30, wherein the first deployment manager is further operable to collect at least one file-asset of the component in the individual export file, wherein each file-asset is configured to operate on the system at the remote location.

72. (Currently amended) The system according to claim 71, wherein ~~designating the component comprises querying~~ the first deployment manager is further operable to query a subsystem.

73. (Previously Presented) The system according to claim 72, wherein the subsystem is configured to manage a type of component, wherein the type includes at least one of a site, module, template and style.

74. (Previously Presented) The system according to claim 73, wherein the subsystem collects each object and each file-asset.

75. (Previously Presented) The system according to claim 37, wherein the second deployment manager is further operable to:

extract at least one file-asset of the component from the individual export file; and store each file-asset to a location on the system at the remote location.

76. (Previously Presented) The system according to claim 75, wherein the second deployment manager further includes a subsystem, wherein the subsystem is configured to manage a type of component, wherein the type includes at least one of a site, module, template and style.

77. (Previously Presented) The system according to claim 43, wherein the component module is further operable to:

extract at least one file-asset of the component from the individual export file; and store each file-asset to a location on the system at the remote location.

78. (Previously Presented) The system according to claim 77, wherein the component further includes a subsystem, wherein the subsystem is configured to manage a type of component, wherein the type includes at least one of a site, module, template and style.

79. (Currently amended) A computer-implemented method of deploying a component of a functioning application between systems, the method comprising the steps of:

designating a component of the functioning application intended for export, wherein the functioning application comprises a collection of software objects manipulatable by a set of users having assigned privileges defined by permissions associated with each software object in the collection of software objects;

collecting a set of objects of the component of the functioning application designated for export in an individual export file;

transferring the individual export file to a system at a remote location; and

extracting each object from the individual export file to a location, wherein each object is configured to operate on the system at the remote location.

80. (Previously Presented) The method of claim 79, wherein the component is a software component used by the application, a configuration of a software component or data used with a software component.

81. (Previously Presented) The method according of claim 80, further comprising instantiating each object that is a non-file asset.

82. (Previously Presented) The method according to claim 80, wherein each non-file asset includes at least one of: a permissions object, a user preference object, a settings object, a menu object, an instantiated programming language object and a user group object.

83. (Currently amended) A computer program product embodied in a computer readable medium for deploying a component of a functioning application between systems in a portal framework, the computer ~~program product comprising: a computer readable medium; and~~ readable medium carrying computer program instructions, ~~recorded on the computer readable medium,~~ executable by a processor, for performing the ~~step~~ steps of:

designating a component of the ~~site~~ functioning application intended for export, wherein the functioning application comprises a collection of software objects manipulatable by a set of users having assigned privileges defined by permissions associated with each software object in the collection of software objects;

collecting at least one object of the component designated for export as in an individual export file, wherein the at least one object includes at least one non-file asset configured to operate on the system at the remote location;

transferring the individual export file to a system at a remote location; and

extracting the assets of the component from the individual export file to a plurality of locations on the system at the remote location.

84. (Previously Presented) The computer program product of claim 83, wherein the component is a software component used by the application, a configuration of a software component or data used with a software component.

85. (Previously Presented) The computer program product according of claim 84, wherein the computer program instructions are further executable for performing the step of: instantiating each object that is a non-file asset.

86. (Previously Presented) The computer program product according to claim 84, wherein each non-file asset includes at least one of: a permissions object, a user preference object, a settings object, a menu object, an instantiated programming language object and a user group object.

87. (Currently amended) A computer-implemented method of deploying a functioning application between systems, comprising:

collecting a set of software components used by a first application on a first system in a set of individual export files, wherein the first application comprises a collection of software objects manipulatable by a set of users having assigned privileges defined by permissions associated with each software object in the collection of software objects;

collecting a set of configurations for use with the set of software components in the set of individual export files;

collecting the data utilized by the software components in the set of individual export files;

collecting the set of individual export files into a group export file;

transferring the group export file to a second system at a remote location;

extracting the set of individual export files from the group export file;

extracting the set of software components, the set of configurations, and the data from the set of individual export files to create a second application on the second system, wherein the second application has the same state as the first application.

88. (Previously Presented) The method of claim 87, wherein the first application is executing.